Is neural computing the key to artificial intelligence?

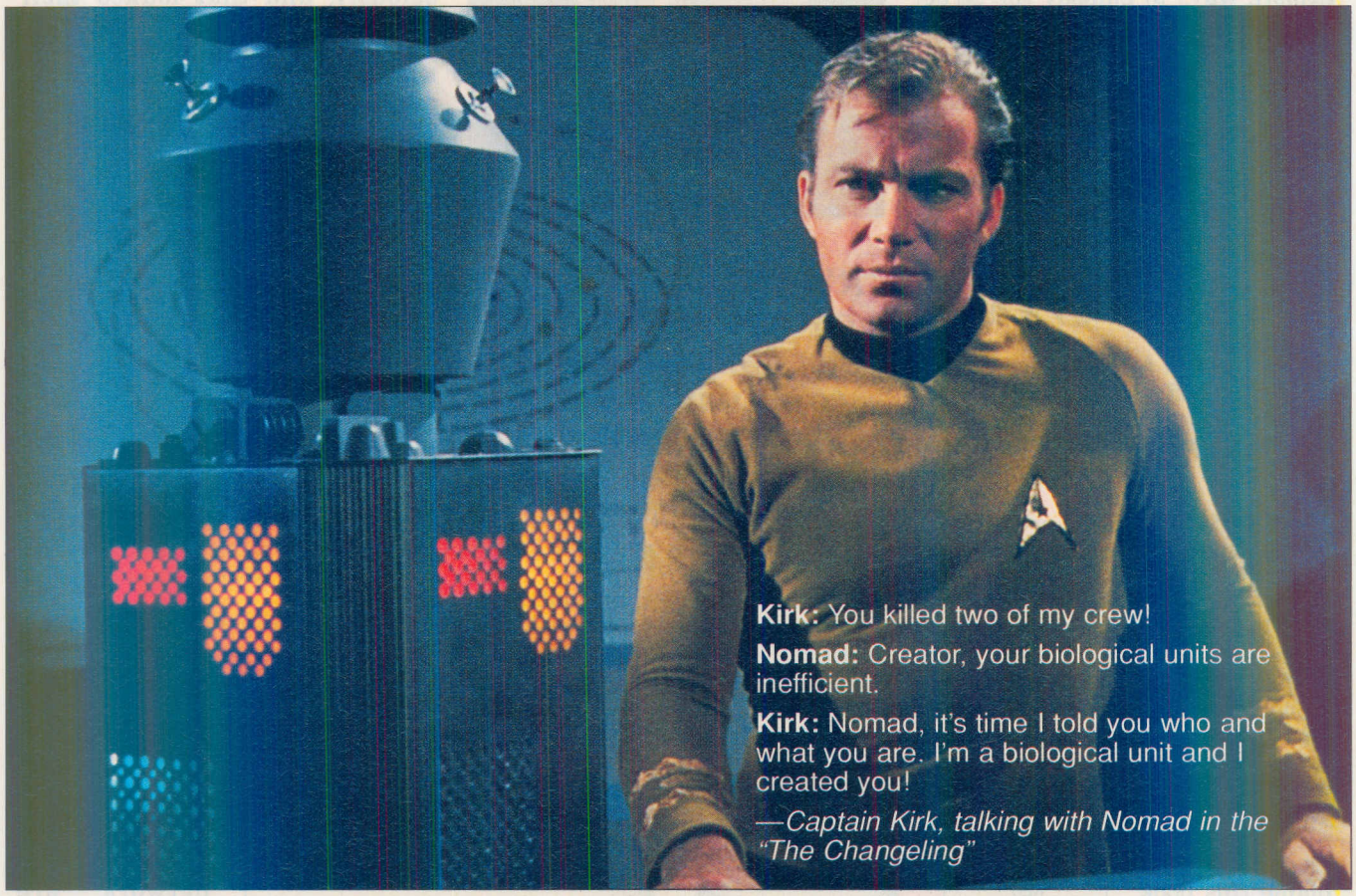


PHOTO COURTESY OF PARAMOUNT PICTURES

Mike Donlin and Jeffrey Child

When people speculate about whether computers can learn to think, evil robots, such as Nomad running amok on the Starship *Enterprise*, often enter the conversation. In the *Star Trek* episode, Captain Kirk talks the sinister Nomad into blowing itself up, but the notion of a computer that can control human destiny makes many folks nervous. Computer professionals scoff at such silliness, even while acknowledging that advances in hardware and software have given computers the ability to emulate some human traits.

People have been fascinated with the idea of a machine that could think since the days of ENIAC, one of the first "electronic brains." ENIAC had 30,000 vacuum tubes and 50,000 relays, filled a large room and could rip through mathematical calculations at a blistering 13 operations a second.

Presently, computer technology resides somewhere between ENIAC and Nomad, but advances in artificial intelligence, and particularly in neural networks, have caused a surge of interest in the thinking power of computers.

Nothing new about neural networks

The concept of neural networks has been around in some form since World War II, but it's only in the last six or seven years that working products have been developed that attempt to "learn" about and predict reality. In their infancy, neural networks and neural computing were the work of theorists who observed similarities in the way that computers and humans think. In both cases, a large amount of information is manipulated by breaking it into small particles—using gates in computers and neurons in humans. Gates handle data by fluctuating between an "on" and "off" state, and neurons do the same by firing (on) or not firing (off).

Scientists have postulated that human thought oc-

curs when two neurons fire simultaneously—and that their connection, called a synapse, is given more weight than would be the case if the neurons were connected but not firing. Because of these similarities between human and computer thought, researchers have begun to explore ways to embody the structures of human intelligence in machines.

In the case of connecting neurons, either through hardware or software, to emulate the brain, the task has been daunting. The human brain contains 100 billion neurons, each connected to 10,000 others by synapses. Building such a complex computer is a ridiculous idea, even with the staggering advances made in computer technology in the last twenty years.

Also, neural networks were dealt a blow in 1969 when Marvin Minski and Seymour Pappert wrote a book called *Perceptrons*, which postulated that neural network research was a waste of time. Minski, one of the founding fathers of the artificial intelligence movement, refused to believe that software could simulate the behavior of human neurons. Minski's vision of artificial intelligence (AI) was far more comprehensive than just neural network technology, and he scoffed at those who wanted to reduce his broad theories to a set of equations that could solve only simple problems. Many experts blame this book for derailing neural network research and encouraging the expert-system theories favored by the authors.

Expert systems, in turn, have fallen out of favor in recent years, because they use a prohibitive amount of computing power to solve problems. Although there are some areas where encoding the skills of an expert and programming a computer to carry them out seem feasible, for most complex tasks the intuition of an expert is simply too difficult to understand or too time-consuming to write out.

"It's really important to understand that neural computing has nothing to do with building brains."

—Casimir Klimasauskas, president of NeuralWare, Pittsburgh, PA

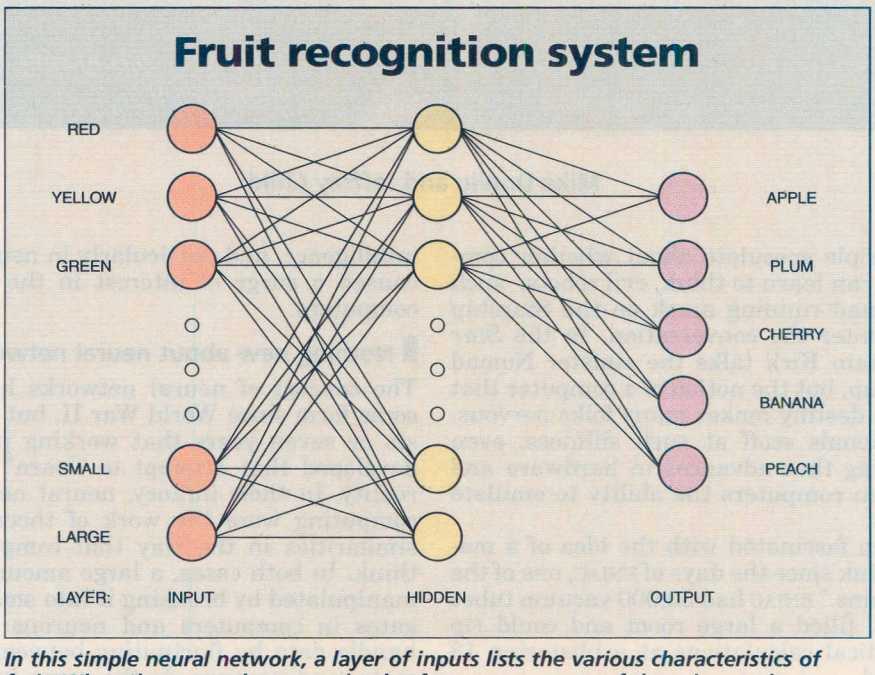
"To make an expert system work, you have to know why the expert makes decisions," says Steve Bissett, senior vice-president at Synaptics (San Jose, CA), a neural network IC firm. "The problem is that most experts can't tell you all the rules that go into their decisions. They use past knowledge and intuition unconsciously, and that makes their knowledge difficult to codify. Even if you could write out enough knowledge to program into a computer, the amount of data would be so large that it would be prohibitively expensive to program and use."

Expert systems are by no means finished, but there's also been a renewed interest in neural computing in recent years, particularly in the fields of signal processing, forecasting and pattern recognition. Even though there are different ways to emulate the neural connection model of the brain, both in hardware and in software, all neural networks share certain common characteristics: they use artificial neurons that are connected to at least one other neuron, and they create their own representations of reality based on some form of learning model.

Learning by example

Fundamentally, all neural networks learn by association. For example, a neural network can learn to identify an apple by associating the inputs "round," "red" and "fruit" with the output "apple." The neurons in a neural network are usually organized in three layers: input, hidden and output. Sometimes more than one hidden layer is used for complex analysis.

There are many ways that neural networks can learn, but the most common way is through example and repetition, also called back-propagation. Each time an input is given to the network ("round," "red"



In this simple neural network, a layer of inputs lists the various characteristics of fruit. When the network gets a stimulus from one or more of these inputs, it responds with an output. If the network guesses incorrectly, it adjusts the correlation of the internal connections or synaptic weights until it gets a correct answer.

or "fruit" from our example), it gives an answer. Naturally, when a network is new, the guess will probably be wrong. But over time, as the network gathers more and more data, it will begin to zero in on the right answer. When it's fully trained, it can deliver an answer that's more or less accurate, depending on the complexity of the task.

Every time the network guesses wrong, it adjusts its internal connections until it gets the right answer. These adjustments are made via synaptic weights, which give relative importance to data as it's applied to a task at hand. These weights can be implemented in hardware or software, and it's this ability to gauge the importance of data that separates neural computing from a purely digital computational process. Although the synapses and weights can be made up of analog circuitry, digital components or software, the weighting procedure makes neural computing appear to have an analog nature—a characteristic that's especially important in performing tasks such as

Neural computing: What it is and what it isn't

Digital computers ushered in the information age, and as they have become smaller, more personalized and increasingly powerful, they've touched more and more aspects of our lives. But as brains for robots, computers still leave a lot to be desired. Recognizing natural objects in the real world, for example, is well beyond the grasp of modern technology. Are we just waiting for more computing power in a smaller package, or are we lacking a fundamental ingredient?

Computers perform logical computation. They operate on precise input information with a programmed sequence of instructions and produce a precise output. Computers are much better than humans at operations such as long division, yet, when it comes to pattern recognition, even insects process information better than the most powerful computer.

Scientists have been studying the brain for decades, trying to understand how biological computation works. Over the last ten years, a number of significant advances have been made and the embryonic field of neural networks has been born.

An artificial neural network is a computational structure similar to its biological counterpart, yet much simpler, even when compared to a very small portion of the brain. Nonetheless, artificial neural networks, simulated with digital computers, have already produced excellent results when applied to some real problems, such as predicting the outcome of horse races or playing backgammon. In many cases these results have been better than the best of the traditional logical or rule-based approaches.

Artificial neural networks perform what might be called intuitive computation. Rather than being programmed with a set of rules, they learn by example; they self-organize. A programmer isn't required to figure out all the rules of the problem—a task that can be extremely difficult—and then write a software program embodying the rules. Rather, neural networks discover the rules for themselves through the process of training. A neural network contains a number of weighted parameters that are modified according to a learning rule, depending on the network's response to training examples.

Much of the research into neural networks has been performed using models built in software, simulating a neural network with a digital computer. While invaluable for research, this approach has limitations in any practical application that requires quick results, because the computer calculates the effect of each interconnection (or synapse) one at a time, whereas nature does it all simultaneously.

Biology, analog or digital?

Nature performs its computational functions using billions of neurons, each having thousands of synapses. To fully exploit the massive parallelism of the neural network structure, it's necessary to develop electrical circuits that approach nature's circuits in density, speed and power dissipation. Unfortunately, digital circuitry falls far short of the mark.

Analog circuits, on the other hand, can perform complex computations by exploiting the physics of semiconductors, which often mirrors the equations governing the behavior of biological neurons. In appropriate structures, this can lead to advantages of 100-to-1 in function density and 10,000-to-1 in power consumption per function, compared with digital circuits.

The set of weights that embodies the learning of the network is stored in floating-gate structures; each analog weight value corresponds to an equivalent amount of electrical charge permanently stored in the floating gate. Synaptics calls this technology adaptive analog VLSI.

Despite their advantages, analog circuits can't match the density and energetic efficiency of nature, but the performance of analog circuits offers such a dramatic leap beyond digital circuits—and the precision of the required computation is so low—that they open up a vast new realm of everyday applications.

Synaptics solves pattern recognition and other problems by combining intuitive computation with logical computation. These two complementary forms of information processing are both essential in solving problems associated with autonomous-intelligent machines, that is, machines capable of naturally interacting with the real world.

***Dr. Frederico Faggin**, co-founder, president and CEO, Synaptics, San Jose, CA*

Neural net chip speeds check reading and verification

Have you ever stood in line at a bank, the supermarket or a gift shop, waiting to cash a check and wondering why it takes so long? Chances are, it has something to do with waiting for a check to clear. Or maybe it's a delay caused by the cashier's inability to read a check.

To address this problem, Verifone (Redwood City, CA), a maker of transaction automation systems, asked itself how you outfit a check reader with enough computing power to handle any check, even those that are difficult to read. Checks in this category include those that are folded, wrinkled or improperly printed. Worse, ink density often varies from one set of characters to another. These variations can cause a great deal of trouble for check readers. Such machines either can't read the checks at all, or they read them inaccurately. Verifone wanted to make a check reader that could read 100 percent of the checks it sees.

Enter Synaptics, a company that was developing an application-specific neural network chip. To make the check reader project feasible, designers at Synaptics reasoned that a simple neural network application would be best suited for its chip. Because there are a limited number of characters on a check, 0 through 9 plus four special symbols, the Verifone application seemed an ideal candidate for a neural network.

Working with Verifone, Synaptics developed an analog neural network chip, the I-1000, specifically for reading checks. The design is historic in that it's the first commercial application using a neural network chip. In fact, the chip is so application-specific that it has a built-in lens through which it reads the images. The lens forms an infrared image on a "retina." The chip then tries to decode the character from this image. Making a number of computations, the chip comes up with an answer. It may see a "9," for example, assigning an 85 or a 95 percent probability to its answer, depending on how good the image is. A perfectly printed character with perfect ink density could receive a 99.5 percent probability. To allow for checks that are printed badly or are wrinkled or folded, Verifone trained the neural network to pass anything with a probability of over 80 percent.

You could find neural computing technology as close as your local gift shop. The Onyx check reader made by Verifone uses an analog-based neural network chip designed by Synaptics. With this chip, the unit can learn what a good check looks like, enabling it to handle a wider variety of checks—even checks that are folded, wrinkled or badly printed. (Thanks to the Forget-Me-Not gift shop in Auburn, CA.)

In an independent laboratory test, the neural-based check reader, dubbed the Onyx, was accurate 99.6 percent of the time. More important, it was capable of reading every check going through it.

Software and hardware support

While Synaptics was primarily responsible for the chip design, Verifone developed software and hardware to support the neural net chip. A 68HC11, Motorola's 8-bit microcontroller, controls all the processes in the machine and interfaces between the I-1000 and the software. While the I-1000 does the decoding, the Onyx also has neural network software. The job of this software is to make sense of what the chip tells it.

If the chip tells it, for example, that there's an 80-percent probability it's reading an "8," the software tries to determine whether the result makes sense. It captures frames to monitor timing as the check goes through the reader. Each frame runs for a different timing interval. The software determines if the acceleration of the frame makes sense; it compares its conclusion to the result produced by the hardware. The final result is based on this comparison.

For memory, the Onyx has 28 kbytes of SRAM. Battery-backed SRAM is used instead of ROM to permit easy updates. If new types of checks are printed, Verifone can simply update (by retraining) the software in the company's lab. The software can then be downloaded to check readers over the phone lines.

Synaptics' neural net chip does more than just read the numbers, however. Using the magnetic properties of the ink used to print the bank number, the neural net can also determine whether or not the check is counterfeit. The ink used has a high iron-oxide content, and since most counterfeit checks are produced using a color copier, they wouldn't show any iron-oxide content. The infrared image indicates iron oxide in the ink based on the frequency of the emission received. If there isn't any iron oxide, no image is recorded, and the check is rejected.

pattern recognition or financial forecasting.

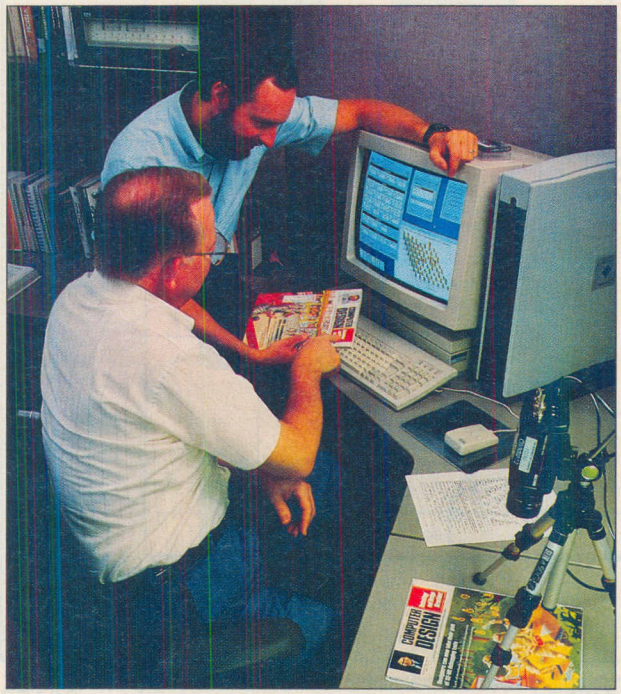
"There's a key difference between neural computing and digital computing," Synaptics' Bissett points out. "Traditional digital computing is like the left-brain or logical thinking that we do. The computer receives a set of rules or programs, then takes input and produces output based on those rules. By definition, it's only as good as the rules that guide it. Neural computing is more like right-brain thinking, which is intuitive. If you wanted a computer to read handwriting, you could try to write rules that would make it recognize an 'S,' for example, and that would be a very linear, but increasingly complex, way to solve the problem. That's not the way we do it. Our circuits are connected in parallel, and compare a shape to previous knowledge to try and categorize it. An important distinction, then, is to try and teach by example rather than programming by rules."

The correlation of neural networks to the way our brains work is what makes them suited to applications that need experiential learning, but is neural computing really thinking? In a word, no. Neural networks are patterned after the architecture of the brain, but in reality their ability to think is far more primitive than that of a common housefly. As a matter of fact, some experts scoff at the notion that neural networks are related to human thought at all, other than in a purely analogous way.

"It's really important to understand that neural computing has nothing to do with building brains," says Casimir Klimasauskas, president of NeuralWare (Pittsburgh, PA). "Neural networks are a collection of mathematical techniques that let you fit formulas to data, curves to data, and group types of data together. Neural networks could have been invented by statisticians, physicists or mathematicians, but the people who invented them were cognitive psychologists and neurobiologists, and so we ended up with the term neural networks. They have nothing to do with brains. I've found that if you try to explain neural networks from a human-thought perspective, people keep trying to fit them into a brain model, and it only confuses them."

Enter fuzzy logic

In spite of such caveats, most people will probably continue to associate neural networks with human thought, particularly because much of the learning process in a neural network takes place in hidden layers or neurons, a processing paradigm similar to human thought. Some researchers are trying to unravel these hidden functions by using fuzzy logic techniques to better understand, or even work in conjunction with, neural networks. Fujitsu (Kawasaki, Japan) is working on a system, for example, that creates a fuzzy rule set based on "if-then" questionnaires that have been filled out by experts. These fuzzy systems are converted into a neural network, which learns about the task and refines its knowledge. Once the neural network has achieved an acceptable degree of accuracy, it's translated back into a fuzzy system so its operation can be analyzed. According to Fujitsu researchers, this model reveals the hidden variables that develop in a neural network. By keeping the fuzzy rule sets and the neural networks as separate systems, the Fujitsu scientists believe they can learn more about how each system works.

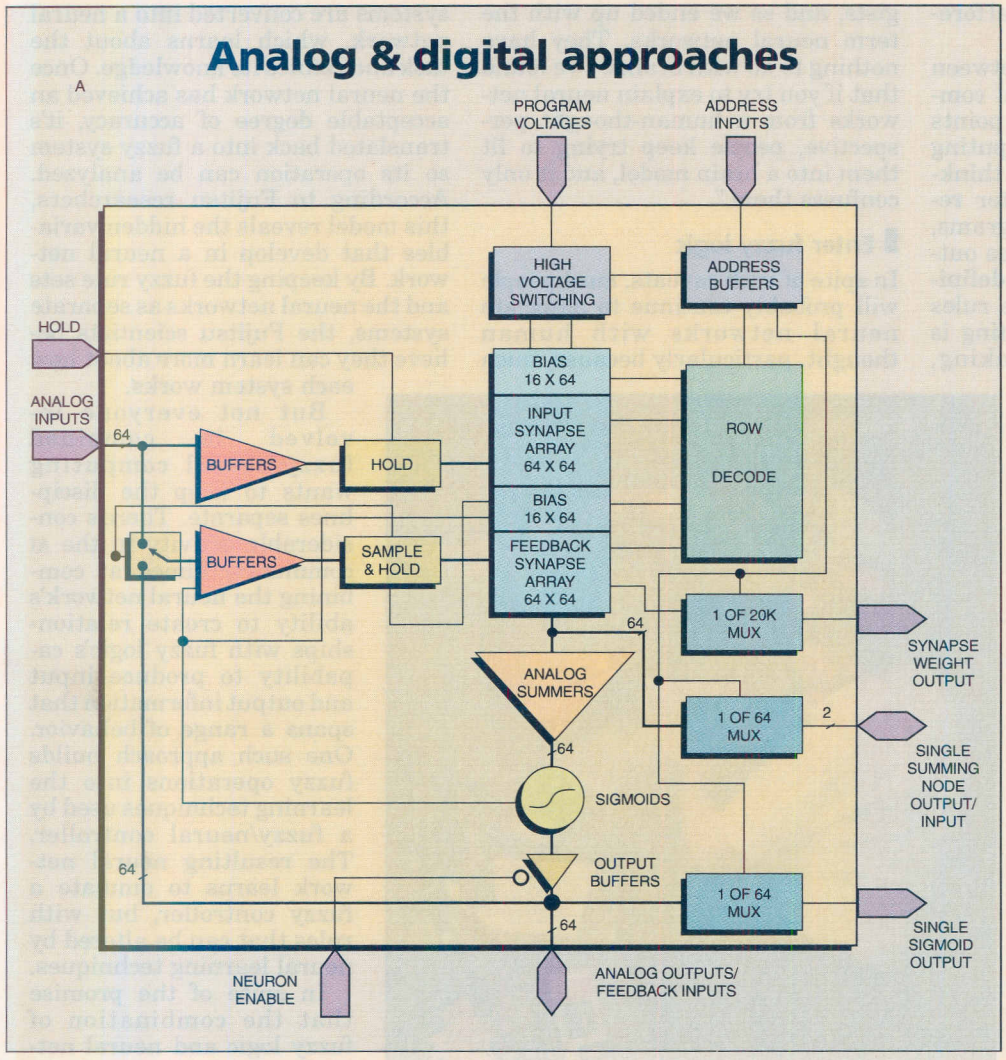


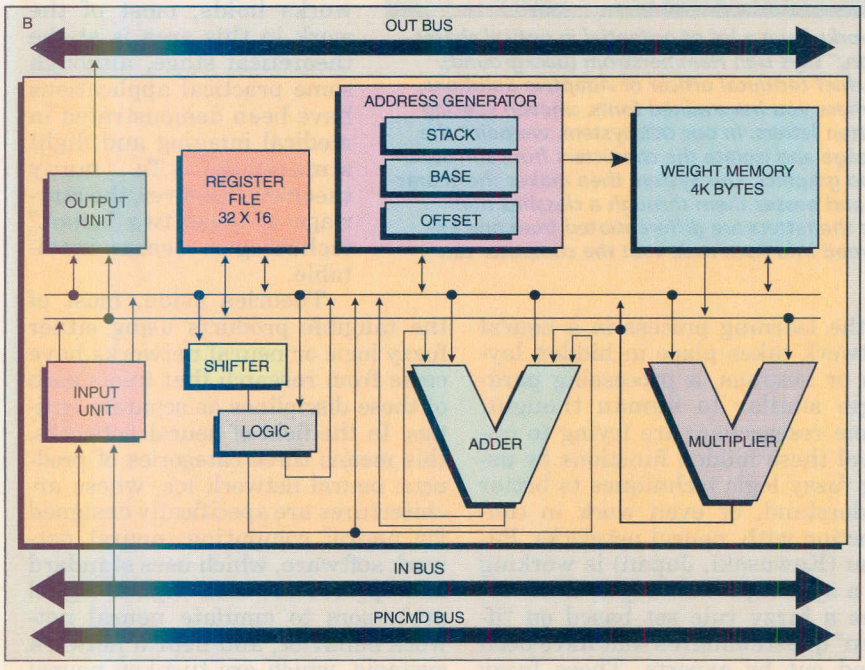
"Neural networks have a lot of potential in optical character recognition," says Dan Hammerstrom (background), founder and chief technical officer of Adaptive Solutions, "especially where you have mixed fonts, uneven spacing and handwritten letters. In our OCR system, we point the camera at a page and isolate the characters from surrounding spaces and graphics. Our system then makes the characters uniform and passes them through a classification phase, where the letters are differentiated from one another and turned into ASCII code that the computer can use."

But not everyone involved in combined fuzzy/neural computing wants to keep the disciplines separate. There's considerable activity in the AI community aimed at combining the neural network's ability to create relationships with fuzzy logic's capability to produce input and output information that spans a range of behavior. One such approach builds fuzzy operations into the learning techniques used by a fuzzy/neural controller. The resulting neural network learns to emulate a fuzzy controller, but with rules that can be altered by neural learning techniques.

In spite of the promise that the combination of fuzzy logic and neural networks holds, most of the work in this area is at the theoretical stage, although some practical applications have been demonstrated in medical imaging and flight simulation. To many theorists, however, the marriage of these two "smart" technologies seems inevitable.

Theories aside, most of the tangible products using either fuzzy logic or neural networks have come from research that treats each of these disciplines as separate entities. In the field of neural networks, this means three categories of products: neural network ICs, whose architectures are specifically designed for neural computing; neural network software, which uses standard microprocessors and digital signal processors to emulate neural network behavior; and neural network systems, which are turnkey neural





The 64 analog inputs to the ETANN device from Intel (top left) are capacitively stored for a limited time by taking the hold pin high. The ETANN's 10,240 synapses store weights as analog transconductance values, each producing an analog output current from an analog input voltage and a stored weight voltage. Currents generated by each of 160 synapses along a dendrite, or ± column pair, are summed to form the net input to a neuron body. The dendrite's sum of currents is then converted to a voltage and passed through a sigmoid function with voltage-controlled gain. The CNAPS-1064 (bottom left) from Adaptive Solutions is an array of 64 processing nodes (PNs), each with 4 kbytes of on-chip SRAM. Each PN resembles a DSP and connects to three global buses: IN Bus, the data input bus; PNCMD, the command bus that dictates the operations of the PN each clock cycle; and the OUT Bus, an output bus.

network computers.

The first commercially available neural network-specific chip was the 80170NX electrically trainable analog neural network (ETANN) device from Intel (Santa Clara, CA). Introduced in 1989, the chip is a 64-neuron, 10,240-synapse IC with inputs organized as two groups, external and recurrent (or feedback); each input contains 80 × 64 synapse arrays.

Intel has also released a development system so that you can simulate, train and operate a high-speed neural network. Dubbed the Intel Neural Network Training System (iNNTS), the package provides two 80170NX devices, two learning simulation software programs, diagnostic software, a programmer interface, an adapter that can run on PC/AT-compatible computers, programming specifications, and full documentation. The iNNTS contains two learning simulation software

continued on page 97

continued from page 92

programs, iBrainmaker and DynaMind. The iBrainmaker program, developed by California Scientific Software (Grass Valley, CA), lets you simulate the network learning process through back-propagation techniques. In back propagation, you present the network model with a data set representing the application problem. Through simulation, iBrainmaker then trains the network to produce a desired response to specific inputs by assigning weights to each of the chip's analog storage elements. Once the network has been trained to solve the application problem, the weights are downloaded or programmed into the ETANN device.

The DynaMind simulation software, developed by NeuroDynamX (South Pasadena, CA), lets you simulate back-propagation learning, but also performs chip-in-loop learning. This technique optimizes the performance of the network by replacing the software simulation of the chip's performance characteristics and specifications with an actual device. The neural network can then "learn around" any minor processing variations occurring in individual ETANN chips.

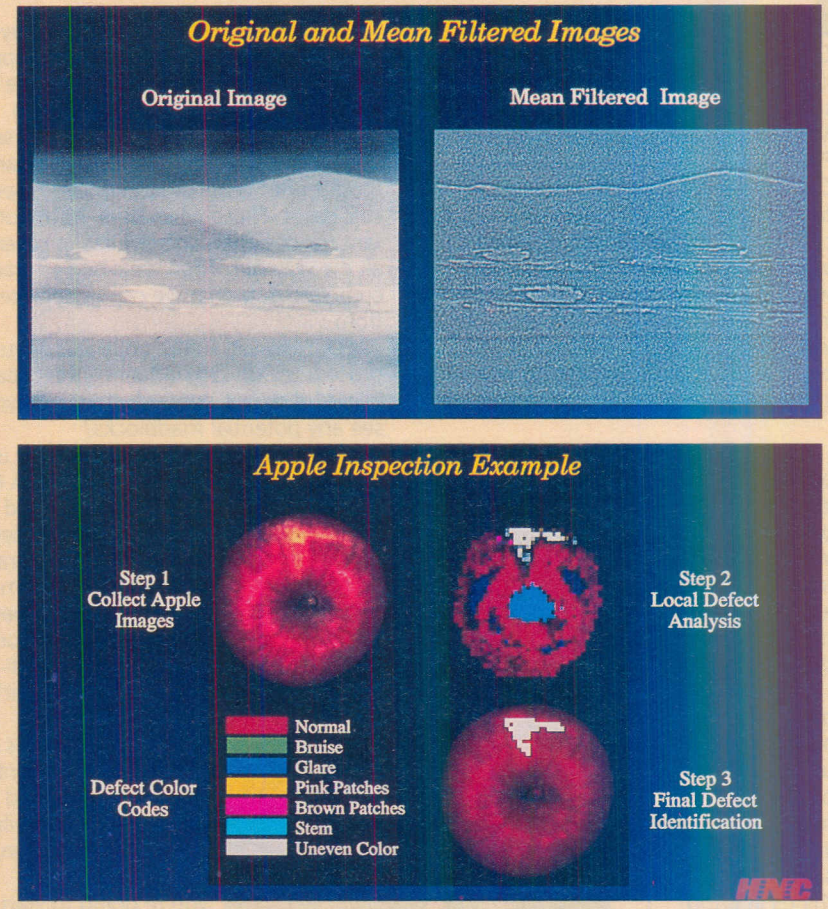
Both iBrainmaker and DynaMind can also be used independently of the Intel development system to create neural network applications.

Fastest-learning chip

A more recent neural network IC is the RN-200, a 256-synapse (16 synapses × 16 neurons) device that Ricoh (Tokyo, Japan) claims is the world's fastest-learning chip of its kind. The device boasts a front-end process of three billion connections per second and a learning speed of 1.5 billion connection updates per second (CUPS) when running at 12 MHz. In Tokyo, Ricoh demonstrated a desktop neural computer system that requires no software. Based on the first-generation RN-100 chip (a one-neuron device with eight synapses), the neurocomputer has a processing speed as high as 128 million neuron connections per second—a figure that Ricoh says will increase when the system incorporates the new RN-200.

One of the first neural network ASICs comes from Neural Semiconductor (Carlsbad, CA). Its CNU3232 has 32 inputs, 1,024 synaptic

Guns and Granny Smiths



In these examples of pattern recognition from HNC, the neural network's ability to learn by association shows the technology to be promising for everything from fruit grading to target recognition for weapons systems.

The apple grading system captures images and feeds them to a neural network, which eventually learns how to determine which characteristics affect an apple's quality. Once the network is trained, the system can classify an apple by comparing its traits to the network's learned base.

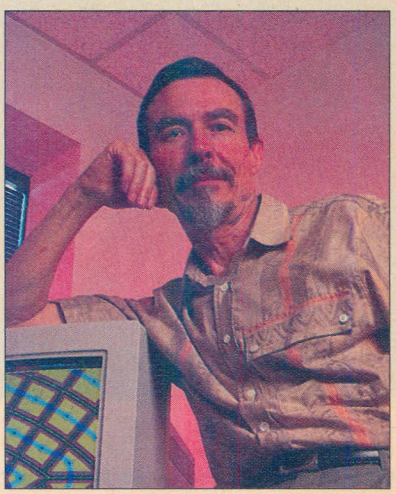
"The tank recognition system," says Ted Crooks, director of customer services at HNC, "was actually an experiment to prove that neural networks could be used in defense systems. Critics of neural networks cited an application where a trained neural system picked out tanks flawlessly until it was discovered that all pictures of tanks were taken in bright light and those without tanks were taken in shade or at night. As soon as those clues were taken away, the network failed. We proved that a neural network could indeed be trained to differentiate a tank's shape from other objects, and that experiment led to a real application."

In this picture, a filtered image of two tanks is outlined so that the network can learn to differentiate them from other objects.

Neural computing challenges the status quo

For this Special Report on Future Computing, Computer Design *interviewed Carver Mead, an expert on the subject of neural computing. Professor Mead is the Gordon and Betty Moore Professor of Computer Science at the California Institute of Technology, where he has taught for 20 years. He's also a co-founder and chairman of Synaptics, a company that develops neural network technology. Mead has pioneered in many areas of electronics, from the invention of the* MESFET *to silicon compilers and, recently,* VLSI *analog neural systems.*

Computer Design: After many years of use in academic circles, neural computing now appears poised to move into practical commercial embedded applications. What's taken so long for this to happen?



Mead: Your question reminds me of how people used to talk about parallel architectures in computers years ago. People said: "We do things in a top-down way." And I'd ask: "Do you really know what the 'top' is?" I'd get these blank looks from people. But today, 15 years later, people are still sorting out what the top is in parallel computing. "Top-down" assumes you know everything in the beginning. That's not a very realistic view of the world. You don't know what's what in the beginning, so you have to evolve your understanding along with the application.

In the neural network business that top-down approach has translated into some rather abortive attempts to make general-purpose neural network chips. Those chips haven't worked well because no one knows what architecture is right for any real application. There are a lot of interesting simulations done in research circles, but none of those are applications that have to work in real time, under real-world conditions.

So then, rather than trying to generalize from basically no information—which is what we're faced with today—it makes more sense to do specific applications. Until you've done that, there's really no royal road to the top to see what is *the* general case. General cases don't come that way. Those come hard-fought after years of working with specific cases. That's why I think its important for designers to solve real problems all the way out. Then we'll begin to accumulate data that we *can* generalize from.

CD: How will future advances in VLSI process technology influence the capabilities of neural network chips? Do you see any potential roadblocks?

Mead: Silicon process technology is very relevant to neural computing. A lot of people have tried to invent brand-new technologies to do neural networks. But it's important to remember that we are riding on the coattails of a silicon technology that's highly evolved. Hundreds of billions of dollars have gone into this most advanced technology that civilization has ever known. And to think that you're going to start from scratch with some other technology and do as well is pretty silly. At Synaptics we've been developing an adaptive analog technology that takes advantage of all the capabilities of process technology. As process technology evolves, it's immediately applicable to this adaptive analog approach. That's not true of other approaches to neural networks.

Transistors as analog devices

CD: As I understand it, your neural net chip uses the transistors in digital semiconductors in an analog way.

Mead: Yes. Transistors are analog devices. Let them be what they are. Digital IC designers have had to work so hard to turn them into 1s and 0s that they lose all that beautiful analog capability. But if you let them exhibit that capability, then you get to save a factor of 10,000 in power consumption, as we did in our chip. It's really remarkable.

The inputs and the intermediate signals are inherently analog signals—they're typically faked out by binary numbers right now in a computer simulation, but that's not the effective way to use them. The effective way is for them to evolve in real time as analog signals. Digital computers not only turn signals into digital values, but they also use discrete time. Those discrete time stamps actually destroy information by aliasing. And because a neural network is nonlinear, there's no theory that tells you how much information you've lost. Using transistors in the continuous (or analog) domain gets rid of those problems. We get back so much for using transistors in an analog way.

Fortunately, the transistors in semiconductors never did know that they were supposed to be digital. They're inherently analog by nature. That means you can use them that way, and the only thing you have to be careful of is that not all the transistors are exactly the same. That's why the adaptive part is so important. Because, not only do you let the neural networks learn from their environment, but they also adapt to changes in the environment. And one of the important changes in the environment is the difference among the transistors. If you do it right, the same paradigm that allows you to adapt to the outside world lets you adapt to differences among the transistors that are on the chip. That lets you use a commercial fab technology to build sophisticated neural net analog processors.

So far, we haven't seen any technological roadblocks in this path. In fact, we're seeing that as the technologies are evolving for digital use, they're actually evolving capabilities that we use in an analog way to make them even more effective.

The analog-digital continuum

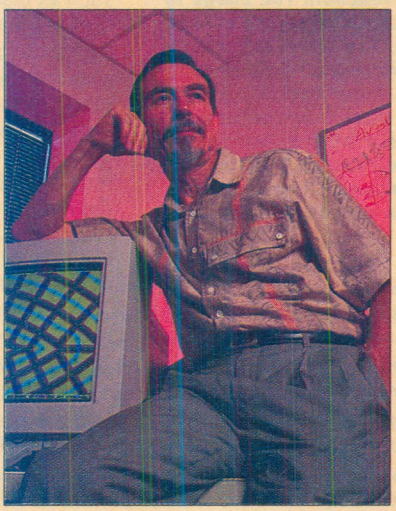
CD: It seems clear that neural computing is not destined to replace traditional logical computing, by any means. It may even open up more opportunities for logical computing. How do you see the situation?

Mead: The real world tends to present you with continuous values: the intensity of a pixel or the value of a waveform, to take two examples. And the digital world deals with discrete symbols: the letters of the alphabet, for example. At some point, the continuous

stuff gets translated into discrete symbols. In the most general terms, this translation process is called classification.

When I started in electronics, almost the whole system was analog. And the classifier was a relay. Then you had a contact closure and that was your digital output. It turned on a light or a heater element in a furnace or something. Now we've gone hard over the other way to where we now have, in today's world, a little bit of antialias filtering and a multiplexer on the analog side. Then there's an analog-to-digital converter, which you could think of as the most brain-dead classifier you can imagine. Because all it does is take analog values and convert them into binary numbers which represent voltage changes. The computer is expected to manipulate these values as if they were numbers and eventually simulate a classification scheme that outputs discrete symbols.

The trend now is toward a much more balanced view of that picture. The analog preprocessing gets the data into a form where it's readily classifiable into appropriate higher-level symbols. If you were doing speech, for example, you'd want to classify the data into phonemes. Our I-1000 chip classifies images directly into character codes. A character code is a lot more meaningful to the computer than the analog values of the pixels. Computers are the way to handle discrete symbols, but they ought to be appropriate discrete symbols.



So we're seeing a return to letting the different technologies do what they're good at. You use an adaptive analog system to do all the preprocessing. You use an analog classifier to decide what the best classification is. That has a digital output which goes into a digital system.

Many of the neural network chips available today are aimed at that classifier job. But they're leaving out all the analog preprocessing that makes the data fit to be classified.

So, I believe that what we're headed for as a field is getting the picture to be more balanced, so the stuff at the analog end of the spectrum is done by adaptive analog technology and the stuff on the digital end is done by digital computers (as it is today). And the classification in the middle, between the two, is done at a level which makes sense for the problem.

The classification problem

CD: I guess you can shoot yourself in the foot if you make the classification problem too complex?

Mead: If you make that too big a problem, then it simply fails a lot. So you want to do those things at an appropriate level. And that's just plain good engineering. So we're not talking about replacing digital computers. It's just a matter of recognizing which technology is most natural for each situation.

CD: Today a lot of neural networks are simulated in software on large, powerful computers.

Mead: Right. It takes too much digital computing for the size of the problem. That limits the applications to the very few, where time isn't critical and you've got the processing power of a Cray computer around. It's only by getting this balance in the technologies that we're going to see very widespread use of the neural paradigm for real systems.

CD: Where do you see neural computing five or 10 years from now?

Mead: If you look at the continuum between analog and digital, we've gone all the way from 90 percent of the system being on the analog side to 90 percent on the digital side. We're headed back toward a balance of about half and half. Over the next few years, you'll see the analog side growing for applications which have to interface to the real world. There's no doubt in my mind that's where neural computing and computing in general are headed.

weights and 32 nodes supporting its activation functions. The one-byte digital inputs and outputs and the weight-storage SRAM are all accessed through an 8-bit I/O bus. Unlike the Intel chip, which uses analog elements, the CNU3232 is a purely digital device that's targeted at embedded system applications.

"We refer to ourselves as a neural ASIC company," says Robert Bagby, president of Neural Semiconductor, "because we expect our customers to build neural network ICs of various sizes, topologies, precisions, and activation functions using our basic architecture. Neural networks are really multiple layers of nonlinear matrix multipliers. We build discrete circuitry for each and every neural multiplier, and place SRAM adjacent to that multiplier to store neural weight values. We also have a neural summation function built into the chip, so you have fully parallel neurons and fully parallel synapses or weights. Because our architecture is purely digital, designs based on it can be manufactured with standard processes for low-cost, high-volume implementations."

The first neural network IC to find its way into a commercially available product comes from Synaptics. The chip, designed for optical character recognition (OCR), hosts an analog sensing array, two neural networks and a digital controller on a single device. The chip is at the heart of a check reader being sold by Verifone (Redwood City, CA). "Until now, if you wanted to perform high-speed OCR, you were limited by the bandwidth between the sensor and the rest of the circuitry," says Synaptics' Bisset. "For most applications using a TV camera, that rate was just 30 images per second. By putting the sensor on the same chip with the classification circuitry, we can do the same task thousands of times per second."

Other solutions

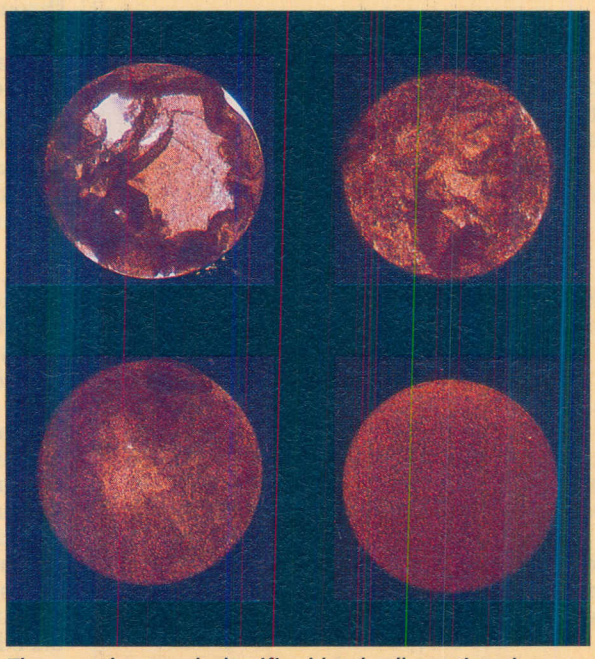
Not all hardware solutions for neural network applications are based on neural-specific silicon. There are systems that use standard digital components for neural computing, as well as for non-neural applications such as Fourier or Gabor transforms. The CNAPS-1064 from Adaptive Solutions (Beaverton, OR) is such a hybrid—an array of 64 processing nodes (PNs), each with its own 4 kbytes of on-chip SRAM. A PN

DSP and neural nets team up for medical research

Any biomedical technician can probably do a better job than a computer of visually identifying reactions in blood cells. But when researchers at the University of California (Davis, CA) needed to classify 72,000 blood-cell reactions each day, and do so accurately and consistently, they moved to an automated approach that combines digital signal processing (DSP) with a neural network simulated in software.

At first glance, the application seemed like a simple image classification problem. Blood-cell images are captured using a high-resolution CCD (charge-coupled device) camera attached to a microscope. Each image is about a millimeter and a half in diameter. That area contains about 100,000 red blood cells. To induce a reaction, antibodies are suspended with the blood cells. After a period of incubation, a reaction may or may not occur. The photos show four degrees of reaction into which the neural network classifies images.

There's added complexity, however, because no two reactions are exactly the same. The network also has to account for rotated views of the exact same image. For example, if a human were looking at the image and felt he or she could better identify the reaction from another angle, the image could simply be rotated. For the computer this would be very complicated. You'd have to show the network how the image looked rotated 5 degrees, and that means the network would have to analyze 72 different versions of the same image. In back-propagation neural nets, the difficulty of training increases proportionally to the number of training examples raised to the third power.



The neural network classifies blood-cell reactions into one of four types. The image in upper lefthand corner shows "no reaction" or a class 0 reaction. The image to the upper right shows some traces of clumping indicating a "weak reaction," or class 1. The lower left image shows even more clumping, indicating a "definite reaction," class 2. And finally, to the lower right the image indicates a "complete reaction," class 3, has occurred. Because these images are so complex (over a quarter of a million pixels), it's important to extract only the most relevant pixel data using DSP techniques. Otherwise, the neural network would require an unrealistic amount of computing power.

Preprocessing needed

"In neural networks, the more work you can do before you hand off the project to the neural net, the better off you are," explains Wasyl Malyj, associate development engineer at UC Davis. With this in mind a preprocessing step was included to extract from the blood-cell image only its most critical data.

To accomplish this, a two-dimensional Fast Fourier Transform (FFT) is performed on the image's pixel data, converting it into the frequency domain. This produces a compact feature vector. Sampling algorithms are applied to these vectors to extract useful information. The goal of the preprocessing is to take a very complex image with upwards of 512 x 512 (or over 1/4-million) pixels and extract from that a few hundred bytes of data. "The preprocessing reduced the amount of stuff that the neural net didn't need to learn, simplifying its structure, its training, and making it possible to implement the neural net with today's technology," says Malyj.

After the image is compressed into a complex feature vector, the vector is fed into the neural net. Cycling this information through the net lets it adjust its connection strengths, and in this way it "learns" to associate particular spectral patterns with particular reactions.

Because of the preprocessing, the input stage of the neural net is typically 128 neurons. To implement the network, the UC Davis researchers developed in software a custom-written back-propagation simulator capable of building nets with three or four layers. The code was written to run on a 486 working in conjunction with a Motorola 96002 floating-point digital signal processor (DSP).

The input layer typically has 128 neurons. The hidden, or middle, layers usually have between 12 and 64 neurons, while the output layer consists of 8 to 20 neurons. Four of the output neurons represent the discrete reaction classes, an additional output provides a confidence metric and the remaining outputs code for a variety of possible error conditions, such as cracks in the plastic tray, bubbles in the sample, shadows cast by bubbles in the mineral oil covering the specimen, lack of blood or reagent in the reaction well, and proper focus.

Training the net

According to Malyj, it took only about six hours to train the neural net. The DSP hardware helped boost its speed. A training set of about 800 images was shown to the neural net, along with the classification under which each image belongs. Once the neural net was trained, technicians began to feed it images that it had never seen before for classification.

The network can be adjusted for various performance levels. "If we tell the neural net to classify absolutely everything, it's accurate to percentages from the high 80s to the low 90s," says Malyj. "That's not as good as a human. But if we tell the net to classify only those images about which it's 'confident,' and to flag the others for us to take a look at, then it classifies about 85 percent of the images at better than 99 percent accuracy."

Researchers can then take the remaining 15 percent of the images and, after they've been scored by a human, use them to retrain the network.

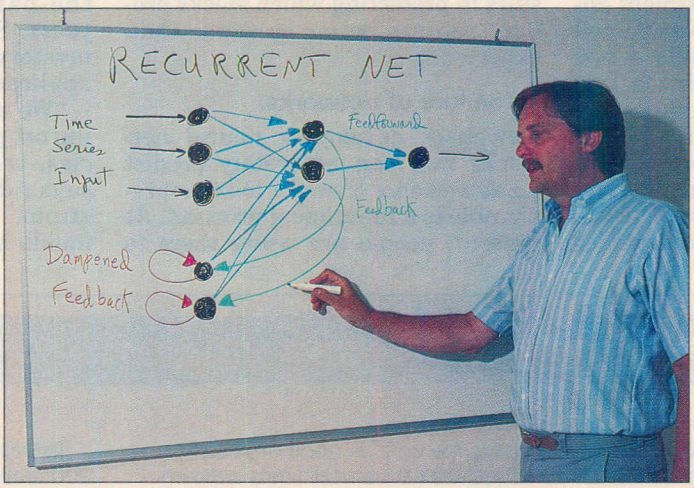
resembles a simple digital signal processor (DSP), and can be programmed for a variety of applications. At 25 MHz, the device can compute 3.2 billion multiply-adds per second. The chip falls somewhere between PC-based software solutions for neural network applications and silicon that's targeted at those applications.

"Given the performance of today's PCs and workstations, you can emulate a lot of neural network applications in software alone," says Dan Hammerstrom, founder and chief technical officer at Adaptive Solutions. "But from a silicon standpoint, there are two approaches you can take. One is a device like ours, which can be customized for a certain domain of applications but is flexible enough to be programmable. The other is a chip like Synaptics', which is designed for a specific task and which may have a price/performance advantage for that application. Both have their place. For complex tasks such as Kanji character recognition, it's simply not feasible to build a custom chip. That's where flexibility pays off."

Software-only solutions

The remaining category of neural network-specific products is software-only solutions, which emulate neural networks on workstations, PCs and mainframes. These programs use either standard microprocessors or DSP chips to perform neural networking tasks, and are available from dozens of companies. The applications are as diverse as the companies that offer them, but most products are being used for pattern recognition, financial analysis and defense-related projects.

In essence, most neural network tasks are based on some form of pattern recognition. Sometimes the sought-after pattern is a visual one—for example, a system that sorts good fruit from bad on a conveyor belt. In this application, the network must be trained to look for a vague trait such as "quality." "The customer who wanted to grade fruit had to train the network to recognize color patterns that distinguished good apples from bad," says Ted Crooks, director of customer services at HNC (San Diego, CA), a neural network software vendor. "By repeatedly presenting data to the network, he trained it to discern the important relationships that define good quality—for example, 'this is premium,' 'this is grade six,' 'this is rotten,' and so on. He didn't define what made up these classifications; he just gave examples of what characteristics are needed to place a piece of fruit there."



"Not all neural networks are created equal," says Steve Ward, president of Ward Systems Group. "A traditional feed-forward network produces one and only one set of outputs from a given set of inputs after it's trained. A recurrent network, on the other hand, may produce many different sets of outputs from one given input set, depending on when in time the input set is presented to the network. Recurrent nets not only learn individual input patterns, but learn sequences in those input patterns as well."

Visual pattern recognition is also being applied to medical research. Some hospitals are using a neural network to sift through hundreds of slides to detect anomalies in blood cells or tissue samples. Early results from these applications show that neural networks achieve a surprising level of accuracy when they're compared to a human performing the same task. As with most applications where a computer equals or bests a human being at a task, fatigue is the deciding factor. Although human expertise may hold the upper hand in picking a cancerous cell out of thousands of normal cells, human fatigue can cancel out some of that expertise, and the capabilities of human versus neural network begin to equalize.

In addition to recognizing flaws in cell structure, some physicians are using neural networks to help them with diagnosis and prognosis. "One of our customers is a neurosurgeon who's using neural networks to predict potential IQ loss after brain surgery," says Jim Blodgett, director of marketing for California Scientific Software. "Traditional statistical methods have been used in the past, but they lose precision in the middle of the bell curve. At each end, either with slight damage or severe damage, the statistics are fine. But in the middle of the curve there can be large variations. Neural networks can use factors such as the severity of an injury or a patient's medical history to make predictions of an operation's outcome."

Financial uses

While neural network research in medicine makes for dramatic reading, there are other applications, particularly in the realm of finance, that might have even greater ramifications. Banks are relying on neural networks to do everything from predicting loan eligibility to spotting credit-card fraud. In the case of loan eligibility, the network is taught to examine the factors that would make a good loan applicant, based on profiles of good and bad loan recipients. The usual criteria are included in the data, such as income, time on the job, credit history, and so on, but neural networks look for unusual patterns or relationships which might escape a human, particularly a human who looks at dozens of applications a day. In the case of credit-card fraud, the network might look for spending patterns which are unusual, such as someone who purchases a wide-screen TV, a trip to Europe and a gourmet meal in a 24-hour period.

The common denominator in these applications is training the network to look for subtle shifts in patterns which are crucial to making a final decision. In the case of stock market predictions, one analyst said that a day on Wall Street is akin to being hit with a dozen data

firehoses at once. The trick is to ascertain which droplets of data in that enormous stream are the ones that will affect the market.

"Naturally, the key here is giving the system data," says Steve Ward, president and technical director of Ward Systems (Frederick, MD). "One of our customers, an investment advisory firm, is using our NeuroShell software to create a Standard and Poors stock market prediction system. The client uses technical indicators to predict short-term changes, and a combination of technical and fundamental indicators to predict long-term changes. The database includes 12,000 companies and looks for fundamental data patterns in areas such as growth and cash flow yield, as well as cross-sectional analysis or valuation change to compute expected return. Early experiments using a model stock portfolio for the years 1987 through 1990 have yielded impressive results in predicting which stocks produce returns better or worse than expected."

The lure of networks

Obviously, developing neural networks that can accurately predict what was once thought to be unpredictable is a tantalizing prospect. At present, people are using them to guess the outcome of everything from the effects of a war on oil prices to the outcome of a horse race. It's true that a lot of these stories sound like hype, especially when they're playing to an audience of computer professionals who've seen their share of flash-in-the-pan technologies over the years. Still, there is an element of mystery to many neural network applications.

"We think there's a lot more going on, particularly in financial circles, than people are letting on," says Adaptive Solutions' Hammerstrom. "After all, if you had the inside story on the stock market,

Neural nets give tin ears a good name

Many applications that are suited for neural computing are tasks at which humans are better. But computers have an advantage over humans. They don't get tired or bored—even after several hours of repetitious work. With this in mind, engineers at CTS (Matamoros, Mexico) made use of a neural network in their loudspeaker manufacturing process.

At its plant, CTS manufactures several million loudspeakers per year. To ensure the quality of the units, a final inspection was performed by a trained operator, skilled at identifying audio defects.

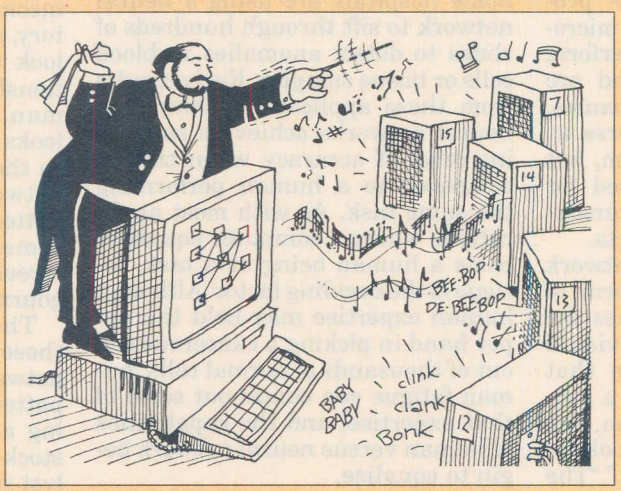
This method had some disadvantages. An operator was required to listen to 2,000 or 3,000 speakers in a single day. Since the final test is very subjective, over the course of a day the operator's fatigue level, emotional state and stress level affected the evaluation. And, because the test is subjective, pass/fail criteria varied from person to person. As a result, many perfectly good speakers were rejected, reducing the factory's yields.

To solve this problem, CTS wanted to remove the subjectivity from the testing process—or at least to make testing consistent from one production run to the next. At first the company used PC-based audio test equipment that was sensitive enough to measure distortions caused by speaker defects.

But this alone wasn't enough. Testing also had to classify the units as good or bad. At first, statistical pass/fail limits were enforced. These turned out to be very unforgiving and depended too much on consistent equipment and fixture setup. In addition, the equipment didn't distinguish among the various types of defects.

It was at that point that Rick Bono, a design engineer at CTS, decided to try using a neural network to classify the results of production testing. A back-propagation neural net was established, consisting of 10 input nodes, 18 hidden-layer nodes and 4 output nodes.

To begin with, a piece of test equipment measures the loudspeakers at different frequencies. The distortion values at these frequencies are the inputs to the neural net. The four possible outputs identify good speakers and three classes of defective units.



To train the neural net, CTS ran over 200 units, both good and bad, through the process, while manually classifying them into one of the four output classes. This required about 20 minutes of computer time on a 25-MHz 486. Once the network was trained, Bono integrated it into the test software using the NeuroShell run-time code generator from Ward Systems Group (Frederick, MD). Network values were developed for each speaker model.

CTS has been using the neural-based system for a year and a half and now runs all its products through this testing process. Over this period of time, the company has seen significant improvements in production yields, thanks to the neural net's ability to classify consistently.

Bono is now considering developing a second-generation system which would incorporate new cases as it works, train invisibly and then implement the new network. This would require a hardware-based implementation of back-propagation, says Bono.

Other upgrades might include adding more output nodes to the neural net. "We've seen different cases pop up—different defect cases that weren't included in the original training," says Bono. "They have distinct patterns, and the net can be trained to recognize them."

Your resource for neural hardware and software

DEVELOPMENT TOOL VENDORS

List courtesy of Martin Middlewood and Tom Schwartz.

★ Adaptive Solutions
CNAPS: Development environment includes CNAPS assembler and library of neural network algorithms.
(503) 690-1236 / Circle 366

AI Ware
N-NET EX: User and program interfaces, functional link net architecture, associative recall. Supervised and unsupervised learning.
(216) 421-2380 / Circle 367

AND America
HNet: Neural-based development system using digital holography principles. Transputer- and PC/Windows-based versions.
(416) 569-0897 (Canada) Circle 368

Applied Cognetics
WinBrain: Develops back-propagation networks. Incorporates multiple transformational models.
(212) 969-8769 / Circle 369

★ California Scientific Software
Brainmaker: Basic version supports up to 512 input neurons, up to six hidden layers. Tutorial and eight sample networks. Imports Excel, Lotus, dBase, binary, and ASIC files. Print/edit neuron matrices.
(800) 284-8112 / Circle 370

EPIC Systems Group
Neuralyst: Integrates neural networks with Excel spreadsheets. Includes macro library for investment analysis.
(818) 564-0383 / Circle 371

★ HNC
ExploreNet 3000: Windows-based application software program for developing and implementing neural network solutions without programming. Database mining program available also.
(619) 546-8877 / Circle 372

Hyperlogic
Owl Neural Network Library: Twenty-four functions for accessing networks supplied as C library. Twenty types of neural networks.
(619) 746-2765 / Circle 373

ImageSoft
ExperNet: Object-oriented tool for creating Windows-based neural networks and knowledge applications.
(800) 245-8840 / Circle 374

Inductive Solutions
NNetSheet: Supports nine algorithms for supervised and unsupervised training. Train network can be ported to a spreadsheet.
(212) 945-0630 / Circle 375

Mathworks
Neural Network Toolbox: Includes learning rules, transfer functions and training and design procedures for implementing neural networks.
(508) 653-1415 / Circle 376

Martingale Research
SYSPRO: FORTRAN-based neural network simulation and prototyping tool.
(214) 422-4570 / Circle 377

Neural Computer Sciences
NeuralDesk: Supports many algorithms. Manual and automatic training of neural networks.
44-703-667775 (UK) Circle 378

Neural Systems
Genesis: Development environment for interfacing neural networks to application software.
(604) 263-3667 (Canada) Circle 379

★ NeuralWare
NeuralWorks: Neural network chip development, open architecture, 8-k back-propagation, makes network types from libraries and creates diagnostic tools.
(412) 787-8222 / Circle 380

Neurix
MacBrain: Flexible neural connections, activation rules, 3-D graphs, interactive modeling, visual macro language.
(617) 426-5096 / Circle 381

★ NeuroDynamX
DynaMind: Train networks on Intel's 80170NX ETANN and Intel multichip board. Can read and store network trained in emulation mode and download weights to chip.
(800) 747-3531 / Circle 382

NeuroSym
Neural CASE: Supports four network paradigms: BPN, CPN, RN, and SOM.
(713) 523-5777 / Circle 383

Peak Software
Autonet: Constructs networks from training data sets consisting of input variables and expected results. Networks may also be created from command line.
(612) 854-0228 / Circle 384

SAIC Artificial Neural Systems
Delta ANSpec: Language for defining and implementing parallel distributed processing systems.
(619) 546-6005 / Circle 385

Software Bytes
ET 2.0: Simulates text, graphics and Windows. Back-error propagation neural networks with Borland C/C++ source code, ET Graphics and Windows slide neural networks on equivalent VGA screens.
(800) 521-4119 / Circle 386

Software Frontiers
Neural Network Toolkit: Development software for neural network applications. C source code included.
(800) 475-9082 / Circle 387

Talon Development
Brain: Lotus 1-2-3 add-in for neural net development.
(414) 962-7246 / Circle 388

★ Ward Systems Group
NeuroShell: Shell program imports ASCII spreadsheet problem files. Example included. Windows version can build up to 128 interacting networks. Supports dBase. Includes run-time option. **(301) 662-7950**
Circle 389

CHIP AND HARDWARE VENDORS

★ Adaptive Solutions
CNAPS System: A 5-billion-connection-per-second neurocomputer. The system has a back-propagation learning rate of 1 billion connection updates per second.
(503) 690-1236 / Circle 390

American NeuralLogix
NLX420: Neural processor slice. A digital chip designed for real-time neural network systems. This 20-MHz device contains 16 processing elements, and can have up to 64,000 16-bit synaptic inputs.
(407) 322-5608 / Circle 391

★ Intel
80170NX ETANN: An electrically trainable analog neural network chip. One chip can perform over 2 billion multiply-accumulate operations per second.
(408) 765-9235 / Circle 392

★ Neural Semiconductor
CNU3232: Digital neural net chip implements a single-layer network of 32 inputs and 32 nodes, capable of processing 100,000 patterns per second.
(619) 931-7600 / Circle 393

★ Ricoh
RN-200: A neural net chip that implements a 256-synapse neural net composed of 16 synapses by 16 neurons. The chip is a 200,000-gate array built on 0.8-µm CMOS. It has a forward process of 3 billion connections per second and a learning speed of 1.5 billion connections per second.
(408) 432-8800 / Circle 394

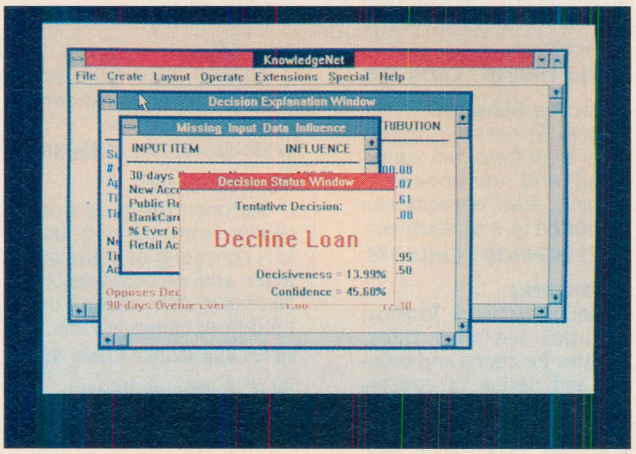
★ Synaptics
I-1000: Analog neural network chip designed specifically for reading checks. A neural network-based image sensor reads the image and a neural network trained to recognize the characters on a check interprets them.
(408) 434-0110 / Circle 395

★ Starred vendors were mentioned in this report

would you boast about it, or quietly use it to get rich?"

Perhaps the only area where more mystery prevails than on Wall Street is in defense-related neural network applications. Most of the people involved in such projects are understandably reluctant to discuss the details of their activities, but it's clear that neural networks are being used for such things as missile guidance systems.

"We became interested in neural networks about six or seven years ago, when other artificial intelligence solutions proved to be too slow for our applications," says Dr. David Andes, research fellow at the Naval Air Warfare Center (China Lake, CA). "As you can imagine, the life of a missile is very short—often only a few seconds. The amount of room in which you have to store a guidance system and its power source is also pretty limited. After all, the purpose of the device is to deliver explosives, not electronics. We got into neural networks because biological brains do the type of computing that we need, and they do it very fast and in a very small area. We're trying to build a guidance system that can hit a target without needing human intervention."



Banks are using neural network software to verify a person's loan eligibility. In this application from HNC, variables such as an applicant's time at a fixed address, time at a particular job, previous loan delinquency, and savings account data are passed through the network for evaluation. The network builds a profile of the loan applicant and makes a recommendation to the financial institution.

Naturally, for these applications a neural network must be trained to recognize a target in the confusion of battle, something that heat-seeking devices find problematic. But trying to give a neural network enough data to find targets in rapidly changing battle situations is a daunting task.

"Neural networks are notorious for picking up on things that you don't want them to," Andes cautions. "We heard about one application where a neural network was picking out the enemy from a visual field with perfect accuracy. Naturally, everyone got suspicious and investigated more closely. It turned out that the system produced a 60-cycle hum whenever the picture of the bad guy was shown, so the network just incorporated that into the equation—if hum, bad guy; if no hum, good guy. They took away the hum and the network was lost."

Neural network's impact

When the breadth of neural networking applications is examined, it's clear that if they're refined, they will affect our lives as much as any technological breakthrough of the twentieth century. But trying to find out where to separate fact from fiction is difficult, particularly when those who are really successful with neural networks are reluctant to divulge too many details. And so the questions remain—can you use a neural network in your job, and how will neural networks ultimately affect your life?

As far as jobs are concerned, neural networks are best suited to analytical tasks that prove too complex or tiring for humans to perform accurately. And certainly because neural networks are based on computer technology, they will affect the electronics industry if they're widely embraced.

Technological applications of neural networks are, in fact, starting to see the light of day. Last February, for example, Intel announced a breakthrough in neural networking, the capability not only to identify patterns but also to read out their locations. The company is using this technology to analyze failures at its fabrication lines. Location information is reported in aeronautics-style polar coordinates. For example, a defective side of a 6-in. wafer may be at 6 o'clock—like a pilot reporting an enemy's position in the sky.

Neural networks could also conceivably guide placement and routing algorithms for chip and printed circuit board design, in essence making decisions intuitively, like experienced engineers, rather than through slavish adherence to algorithms. The potential is only limited by your imagination and by the amount of data that you have.

"When we qualify potential customers' applications, we go by how much organized data we have on them, not on the size of their checkbooks," says HNC's Ted Crooks. "There are plenty of companies out there who have all this data lying around. With a little ingenuity we can turn a large database that used to be a nuisance into an asset. But without that data we don't even bother. All we'd get would be an aggravated customer."

It's true that we are a society awash in data. The government and the banking industry alone probably have enough statistics about most of us to predict our voting habits, buying patterns and loan eligibility.

If all of this sends an Orwellian chill up your spine, you're not alone. Although it's far too soon to predict whether neural networks are another headline-grabbing AI story or the beginning of a computer revolution, one thing is clear. The photographs that accompany this article are taken from real applications. Somewhere a computer might be grading the apples that you eat. It may be deciding whether to give you a loan or recommend you for a job interview. And because most of you reading this article are familiar with what computers can and can't do, you're either smiling right now—or feeling a little queasy. ■